Software Requirements Specification Documentation

for

ICDE-ScholarHub

Version 1.0

Prepared by :
Zerui Wang
40177315
Ding Li
40160073
Jun Huang
40168167

last editing: November 3, 2021

Contents

1 Introduction

1.1 Purpose

We define a system named **ICDE-ScholarHub** which is a web-based Paper Search Engine with ICDE concepts implemented.

The system will practice the idea of ICDE to help the researchers work more efficiently regarding user operation history, teamwork, and paper trending.

Furthermore, the system exposes the API of ICDE for the development of 3rd-party applications or plugins.

1.2 Sytem Problem Statement

During the daily research work, it always happens that researchers forget some critical paper they have read on the web-based paper search system a long-time ago. It takes time to find the critical one from the huge amount of paper list from a different academic area.

More sadly, people may miss the latest paper which is important to their research. Researchers have supervisors, colleagues, and teammates. They like to share the latest paper in the same area with the team. But they do not want to disturb others by sending papers through email too often and it is not well organized.

If there is a system that implements ICDE API, the reading history will be well recorded. The system will log down every activity the user performed.

For example, the researcher can develop a small tool to log down his paper reading history; the research team can develop a tool for sharing team member's paper reading history, paper marking history, area preference, and so on.

The aim of using ICDE-ScholarHub on the paper search system is to help the researchers and their teams work in a more efficient way.

1.3 References

ICDE: `https://www.researchgate.net/publication/220690558_Essential_Software_Architecture_2_ed`

2 Overall Description

2.1 User Scenario & User Story

The system has three main scenarios, Table 2.1, table 2.2, and table 2.3 present three user scenarios for better understanding of the system. And their corresponding user stories are given following the table.

Scenario One

Researcher
Jack is a new university student. He got an individual assignment last week from one of the courses which require academic research. With the assignment demand, Jack should search at least 3 to 4 academic papers for investigation. So he heads to the **ICDE-ScholarHub** website.
He goes to the website and registers an account of it by filling in all the information required by the website.
After finished registering, he then type some keywords of the subject from the assignment. The website displays all the results that match the keywords.
He can read some information about the paper on its detail page. He will know the authors, the year of the publication, the academic area, and the abstraction. He will know that how many people are like this paper, what are the comments of it.
Then he clicks the ”download” button and then gets the pdf file of this paper.
After finish the reading of it, he then login to the website, writes a comment about it and clicks the ”dislike” button to express his idea of this paper.

Table 2.1: New user’s story

1 Jack wants to register a user account, so he can log in to the system.

2 Jack wants to log out from the system, so he would not leave my personal information on the system.

3 Jack wants to search papers with certain keywords, so he can browse the result for his assignment.

4 Jack wants to see the detail page of the paper that he clicks into, so he will know more about the paper.

5 Jack wants to comment on the paper, so he can express his idea of it.

6 Jack wants to download the paper, so he can read the pdf file of the paper.

7 Jack wants to click the ”dislike” button, so the other will know it.

Scenario Two

Researcher
Lucy is a Ph.D. student researching a program for several months. She likes to use the system for paper searching. After work, she will check the paper trending list once to twice a season. With this trending list, it is easy to notice what are the most popular topics and papers people like during the past season. She will know how many people have visited the detail page of the paper. She will know how many new ”like” have been given for the paper from other users. And when she is at work, she can see her paper browsing history on the website and then find the one she wants to read again.

Table 2.2: Common user’s story

1 Lucy wants to see the paper trending board, so she can know how is it like of current academic trend.

2 Lucy wants to see her paper browsing history, so she can retrieve her previous work.

Scenario Three

1 Tom wants to create a research team on the system, so he can share information with other teammate.

2 Tom wants to add Kail to the research team, so they can share more information with each other.

3 Tom wants to remove a teammate, so the whole team will not get the information from this person.

4 Tom wants to share a paper with other teammates on the system team page, so they can get it ASAP.

5 Kail wants to join Tom’s team, so he can work with the team.

Research Team
Tom and **Kail** are teammates within the same research team. They are working on the same project with other professors and schoolmates. The whole team is on the same team created by the ScholarHub system. Tom is the administrator of the team. He can add or remove a member to the team's member board. During their work, Tom can see other team members' activities such as what paper are they currently reading or what is the attitude of the member to a particular paper. Also, every member of the team can share the paper they think is helpful for the team's research so that the others will know about this paper.

Table 2.3: Team users' story

2.2 System Requirements Specification

Top SRS

To fulfill the software descriptions, the system shall have the following SRS:

SRS 1. The system shall have the general user information management module design for the academic researchers;

SRS 2. The system shall have the paper searching and management module and store a massive volume of paper files;

SRS 3. The system shall have a user-paper interacting module for users to upload, download, rate, comment and collect certain paper;

SRS 4. The system shall capture all user paper paper-relevant operations and store those records solid and provide statistical data to users themselves;

SRS 5. The system shall provide the API portal to the 3rd-party developer for accessing records described in specification 5;

SRS 6. The system shall allow users to create, join, leave a research team and provide a role system for team management;

SRS 7. The system shall use the ICDE API to implement features for sharing paper-relevant operations within the same team;

Sub SRS

Regarding SRS 1, the system should have the following modules and features for user information management:

SRS 1.1 User Infomation Module

1 register: ;

2 update information;

SRS 1.2 Login Module

1 login;

2 logout;

Regarding SRS 2, the system should have the following modules and features for paper search service:

SRS 2.1 Paper Download Module

1 download paper;

2 get download records;

SRS 2.2 Paper Operation Module

1 comment paper;

2 like & dislike paper;

3 share paper;

4 get paper operation history;

Regarding SRS 3, the system should have the following module and features for paper operation service:

SRS 3.1 Paper Search Module

1 search paper;

2 view paper detail page;

Regarding SRS 4, the system should have the following modules and features for user's activity capture service:

SRS 4.1 ICDE Record Capture Module

1 capture user's searching keywords;

2 capture user's clicking on certain paper's detail page;

3 capture user's like & dislike operations;

4 capture user's commenting operations;

5 capture user's sharing operations;

SRS 4.2 ICDE Record Access Module

1 get user's searching keywords;

2 get user's clicking on certain paper's detail page;

3 get user's like & dislike operations;

4 get user's commenting operations;

5 get user's sharing operations;

Regarding SRS 5, the system should expose API portal includes following features:

SRS 5.1 Non-user-authorized API:

1 get search trending;

2 get click-rate trending;

SRS 5.2 User-authorized API:

1 get paper search records;

2 get paper rate records;

3 get paper download records;

Regarding SRS 6, the system should have the following module and features for team management service:

SRS 6.1 Team Management Module:

1 create team;

2 join team;

3 view joined team list;

4 leave team;

5 transfer leadership;

6 invite new member;

7 get team list;

Regarding SRS 7, the system should have the following module and features for team member activitiy sharing service:

SRS 7.1 Team Share Module:

1 get other member's like & dislike activities;

2 get other member's click into paper's detail page activities;

3 get shared paper from other members;

2.3 System Architecture Design

The **context diagram** of the system is shown in: Figure 2.1, which indicates that the system has no dependences on outer systems.

The **system architecture diagram** is shown in: Figure 2.2.

The system will follow the MVC architecture since the client end of the system will be web browser. And obviously, it follows the layer pattern.

Notice that component with red box is designed for future development of this system itself. It might not be implemented within the scope of the class project but it will still have interfaces defined.

2.4 System Modeling & Design

2.4.1 Use Cases Diagram

Overall Use Cases: The overall use case is shown as Figure 2.3.

Use cases Diagram: According to the various sub-requirements level, we specify the use cases as follows of user requirements and system requirements level, which are shown in different subgraphs of Figure 2.4 and Figure 2.5.

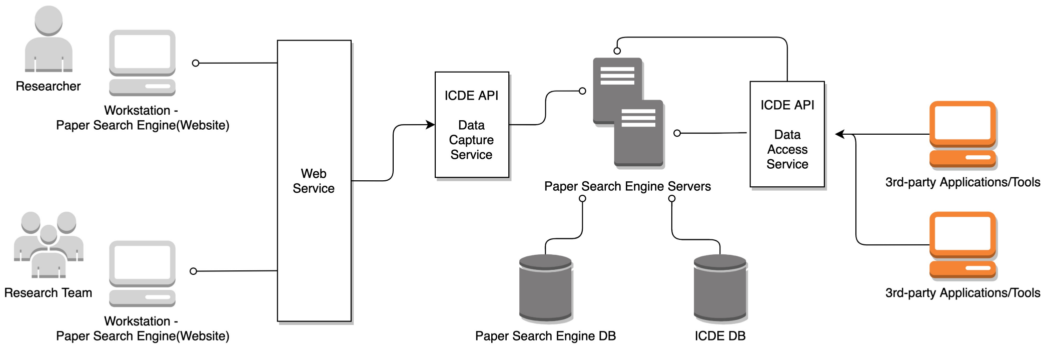

Researcher
Workstation -
Paper Search Engine(Website)
Research Team
Workstation -
Paper Search Engine(Website)
Web
Service
ICDE API
Data
Capture
Service
Paper Search Engine Servers
ICDE API
Data
Access
Service
3rd-party Applications/Tools
3rd-party Applications/Tools
Paper Search Engine DB
ICDE DB

Figure 2.1: System Context

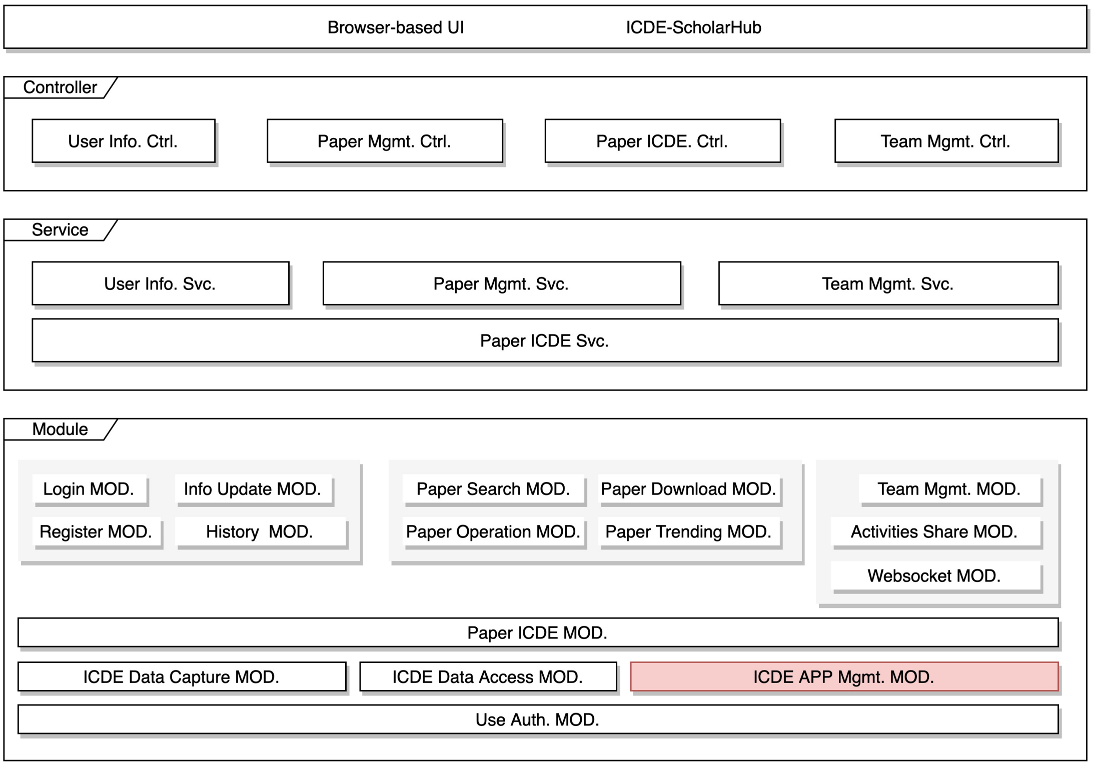


Figure 2.2: System Architecture

2.4.2 System Business Objects Class Diagram

According to the latest system requirement specifications(SRS), the system will have the following bussiness objects(BO) which serve the MVC.

Figure 2.6 shows the very basic BOs of the system. **Record** will be the topmost general BO above any other BO. **Researcher** will represent the solid system user. **PaperMetadata** will represent paper informations. More fields of PaperMetadata can be refer to: `http://bib-it.sourceforge.net/help/fieldsAndEntryTypes.php`.

Figure 2.7 shows the BOs which represent the users' preference about certain paper. **PaperComment** will be the comment wrote by the user. **PaperLikeAndDislike** will represent the preference from the user.

Figure 2.8 shows the BOs which represent the users' paper collection. **PaperCollectionCat** will be the collections which are created by the user. **PaperCollectionRecord** will

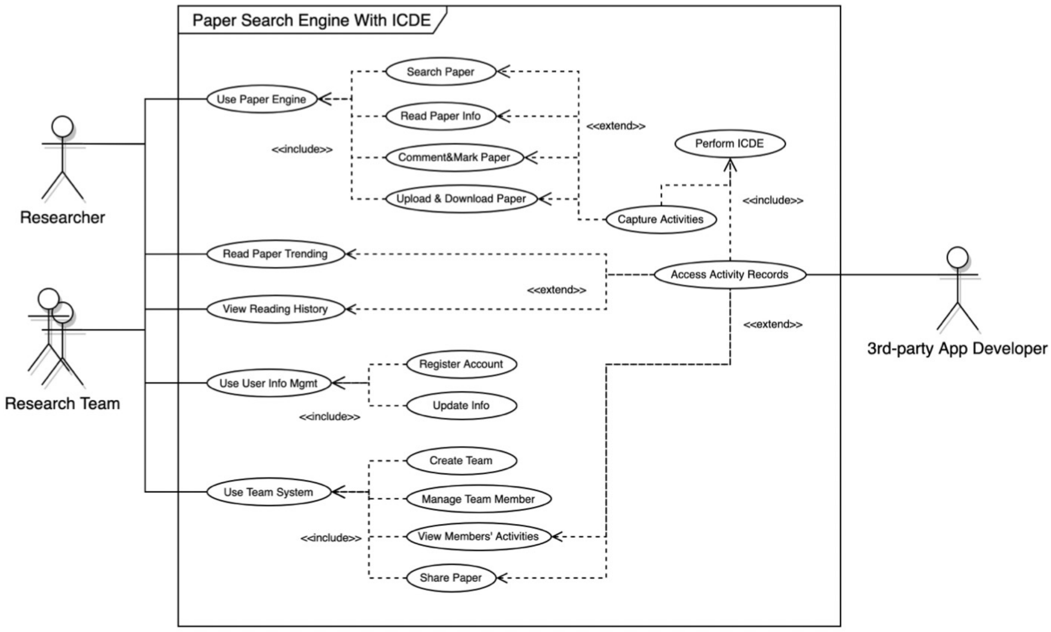


Figure 2.3: Overall Use Cases

represent the single paper that collected within certain collection category.

Figure 2.9 shows the BOs which will gain the benefits from the ICDE. **UserOperationRecord** will be the records that are created by the system whenever the system user performs paper-relevant actions. **PaperClickTrending** will represent the trending of how many clicks on certain papers, this data will generate by the system using the data recorded by ICDE. **SearchTermTrending** will represent the trending of the hottest search terms.

Figure 2.10 shows the BOs which represent Team features of the system. **ResearchTeam** will represent a research team. **ResearchTeamAuthRecord** will represent the relationship between users and teams.

Figure 2.11 shows the BOs which will serve the 3rd-party ICDE application. **ICDEThirdPartyApplication** will represent a registered 3rd-party ICDE application. **ICDEThirdPartyAppAuthRecord** will represent the relationship between ICDE applications and users.

Notice that those BOs are designed for future development of this system. It might not be implemented within the scope of the class project but it will still have classes or interfaces defined.

2.4.3 Diagrams over Representatve Functions

For SRS1.1.1-Register, it can be represent as activity diagram shown in Figure 2.12.

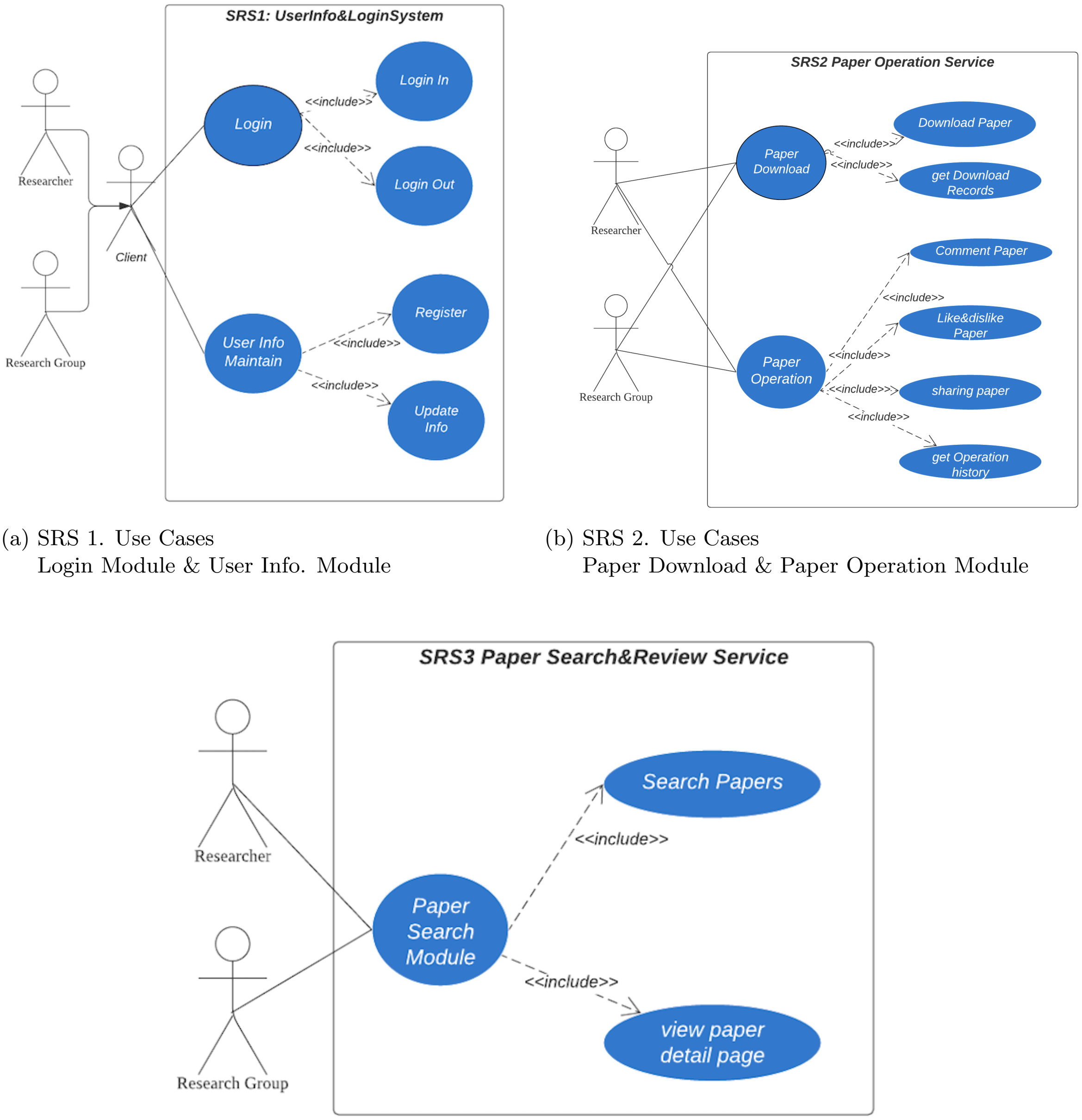
For SRS1.1.2-Update User informations, it can be represent as activity diagram shown in Figure 2.13.

For SRS1.2.1-Login, it can be represent as activity diagram shown in Figure 2.14.

For SRS2.1.1-Download Paper, it can be represent as sequence diagram shown in Figure 2.15.

For SRS 2.1.2, SRS 2.2.4, SRS 4.2, and SRS 5, they can be represent as activity diagram shown in Figure 2.16.

For SRS 6.1.2-Team Invitation, it can be represent as activity diagram shown in Figure



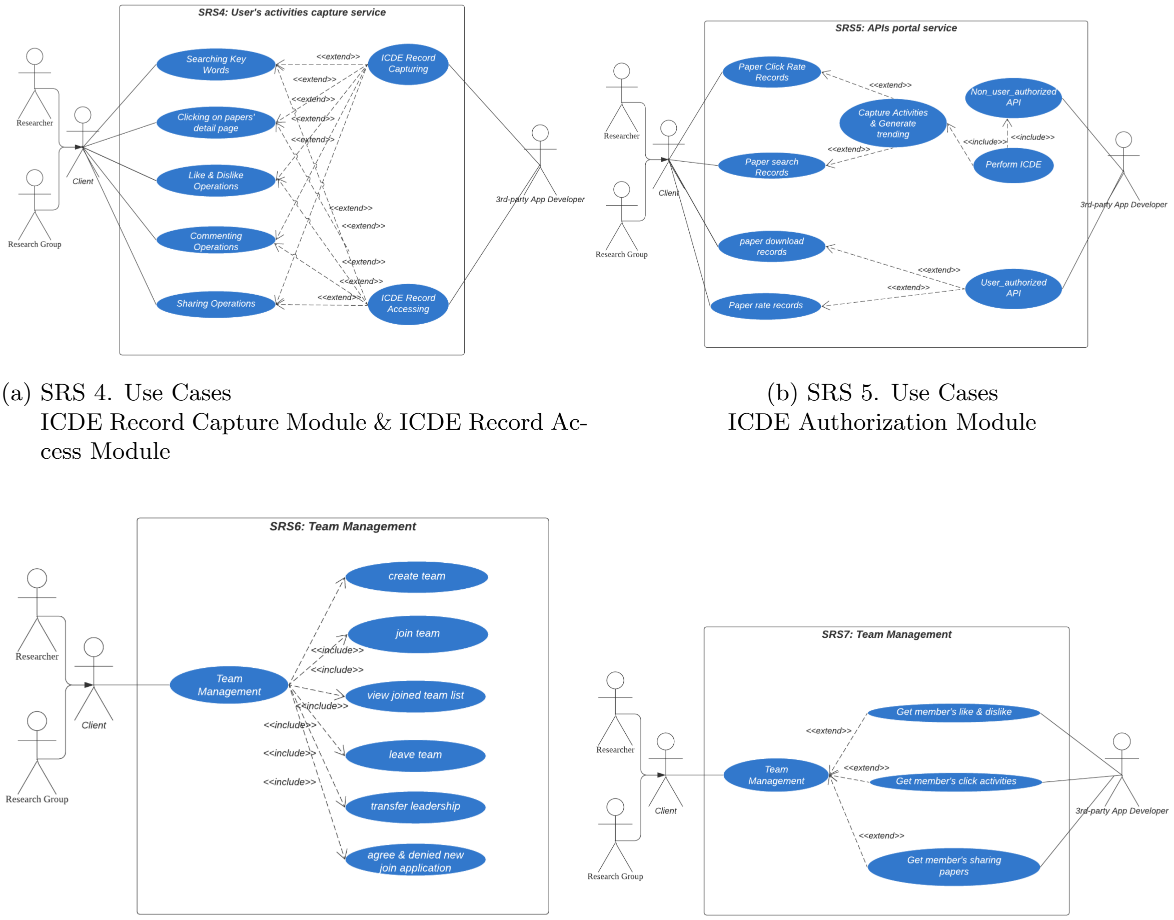
(a) SRS 1. Use Cases
Login Module & User Info. Module

(b) SRS 2. Use Cases
Paper Download & Paper Operation Module

(c) SRS 3. Use Cases
Paper Search Module

Figure 2.4: Use case Diagrams for SRS 1, 2, 3

2.17. Also it indicates how ICDE functions. And its sequence diagram is shown in Figure 2.18. And also its state diagram shown in Figure 2.19



(a) SRS 4. Use Cases
ICDE Record Capture Module & ICDE Record Access Module

(b) SRS 5. Use Cases
ICDE Authorization Module

(c) SRS 6. Use Cases
Team Management Module

(d) SRS 7. Use Cases
Team Share Module

Figure 2.5: Use case Diagrams for SRS 4, 5, 6,7

2.5 Operating Environment

Operating environment for the system is listed in Table 2.4.

System Mode	Database	Back-end	Front-end
B/S Mode	MySQL 8.0	Python 3	Vue 3

Table 2.4: Operating environment

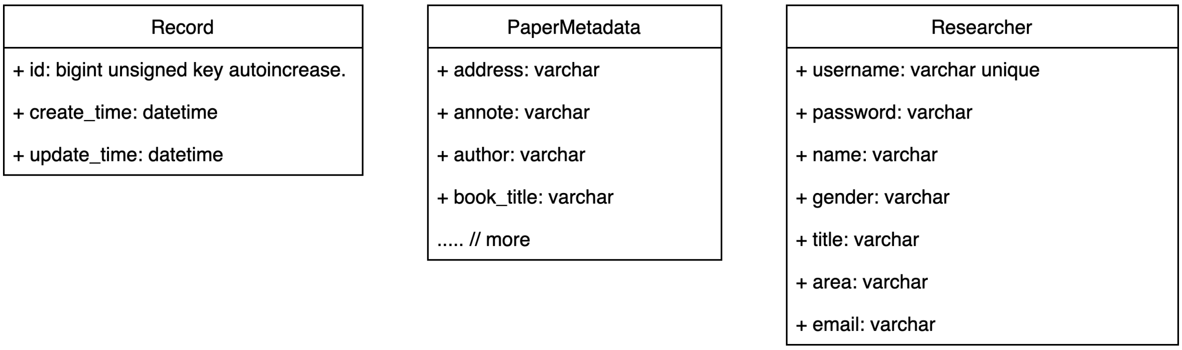


Figure 2.6: Bussiness Objects Part1: Record, Researcher, PaperMetadata

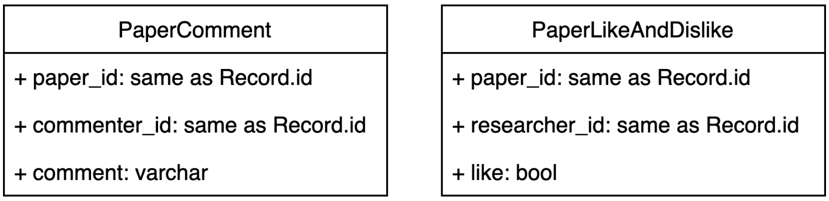

PaperComment
+ paper_id: same as Record.id
+ commenter_id: same as Record.id
+ comment: varchar
PaperLikeAndDislike
+ paper_id: same as Record.id
+ researcher_id: same as Record.id
+ like: bool

Figure 2.7: Bussiness Objects Part2: PaperComment, PaperLikeAndDislike

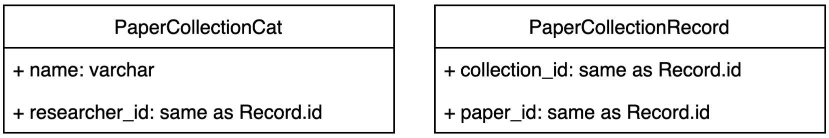

PaperCollectionCat
+ name: varchar
+ researcher_id: same as Record.id
PaperCollectionRecord
+ collection_id: same as Record.id
+ paper_id: same as Record.id

Figure 2.8: Bussiness Objects Part3: PaperCollectionCat, PaperCollectionRecord

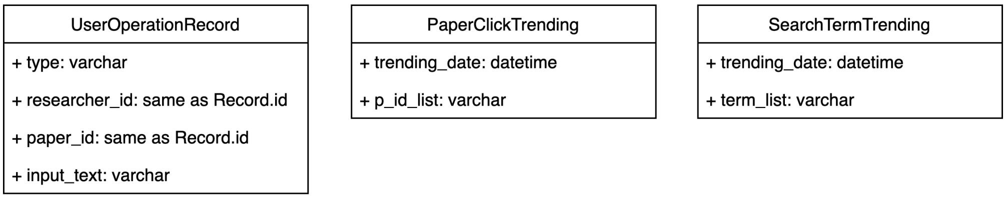


Figure 2.9: Bussiness Objects Part4: UserOperationRecord, PaperClickTrending, SearchTermTrending

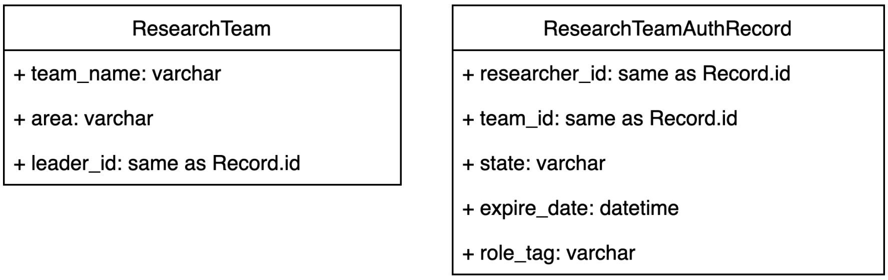


Figure 2.10: Bussiness Objects Part5: ResearchTeam, ResearchTeamAuthRecord

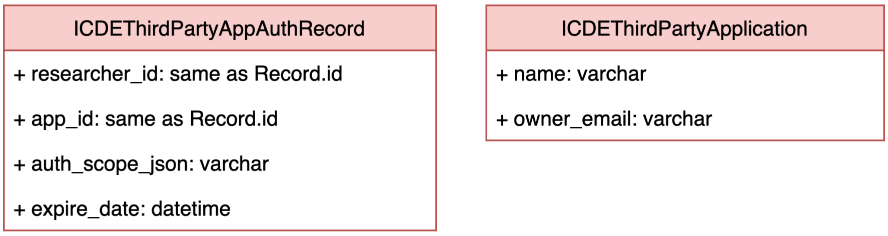

ICDEThirdPartyAppAuthRecord
+ researcher_id: same as Record.id
+ app_id: same as Record.id
+ auth_scope_json: varchar
+ expire_date: datetime
ICDEThirdPartyApplication
+ name: varchar
+ owner_email: varchar

Figure 2.11: Bussiness Objects Part5: ICDEThirdPartyApplication, ICDEThirdPartyAppAuthRecord

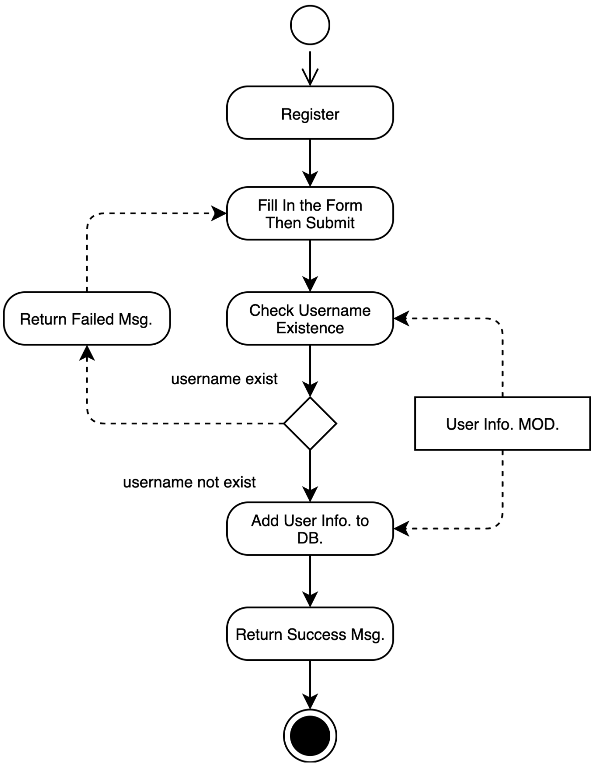


Figure 2.12: Activity Diagram for SRS1.1.1

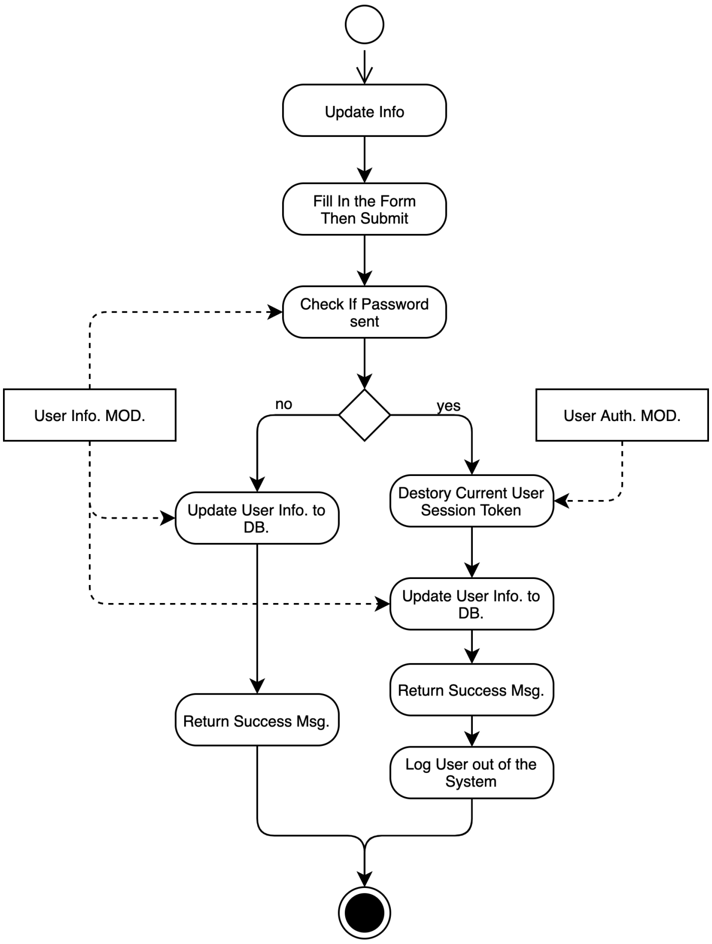


Figure 2.13: Activity Diagram for SRS1.1.2

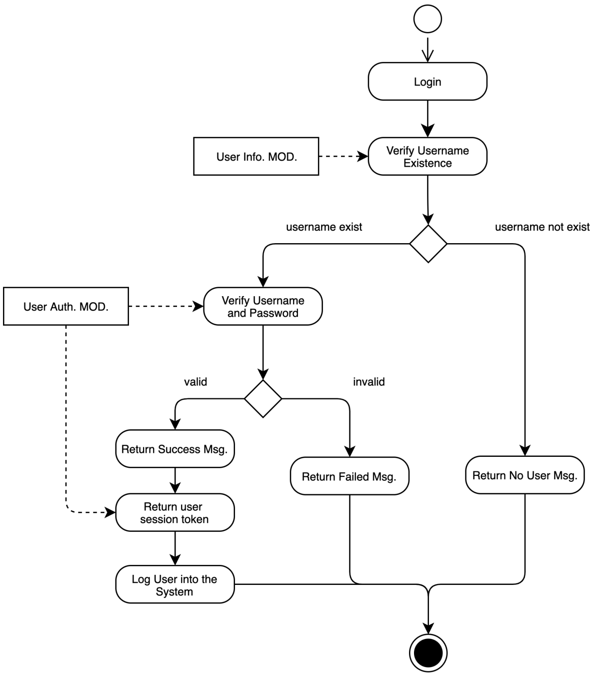


Figure 2.14: Activity Diagram for SRS1.2.1

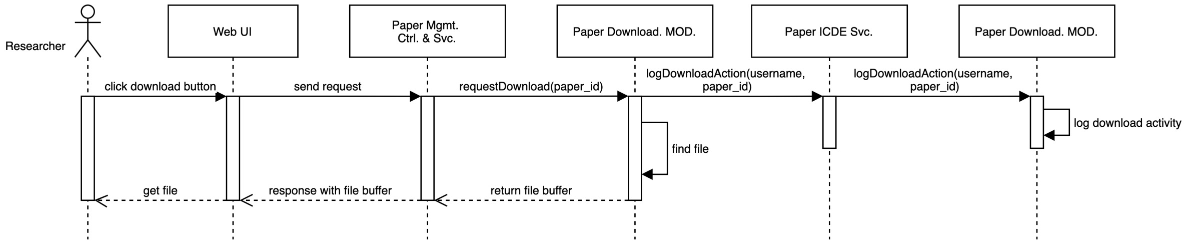


Figure 2.15: Sequence Diagram for SRS2.1.1

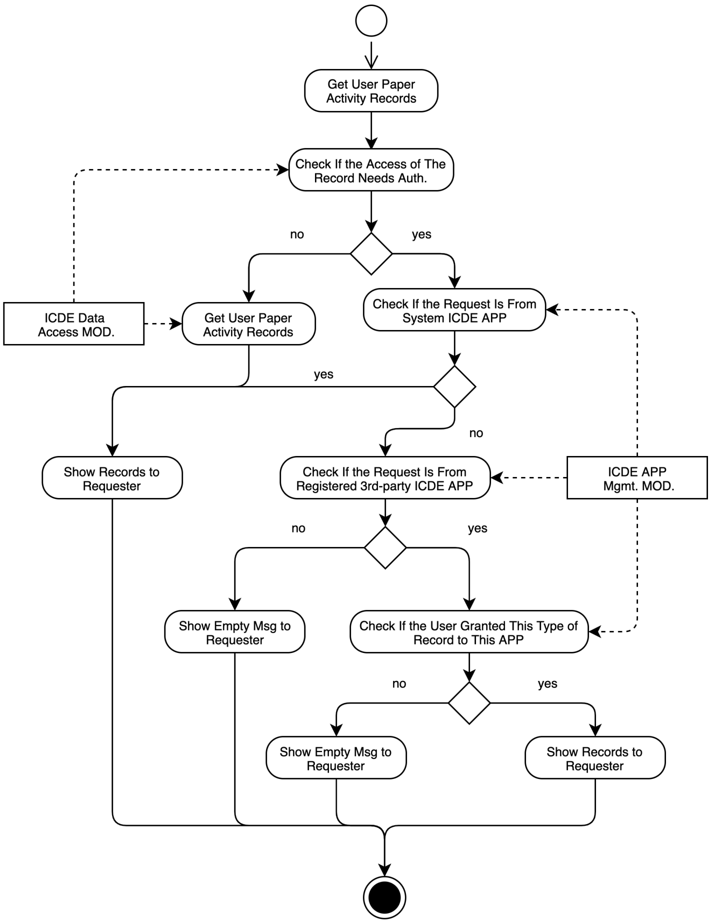

Get User Paper Activity Records
Check If the Access of The Record Needs Auth.
no
yes
ICDE Data Access MOD.
Get User Paper Activity Records
Check If the Request Is From System ICDE APP
yes
no
Show Records to Requester
Check If the Request Is From Registered 3rd-party ICDE APP
ICDE APP Mgmt. MOD.
no
yes
Show Empty Msg to Requester
Check If the User Granted This Type of Record to This APP
no
yes
Show Empty Msg to Requester
Show Records to Requester

Figure 2.16: Activity Diagram for SRS 2.1.2, SRS 2.2.4, SRS 4.2, and SRS 5

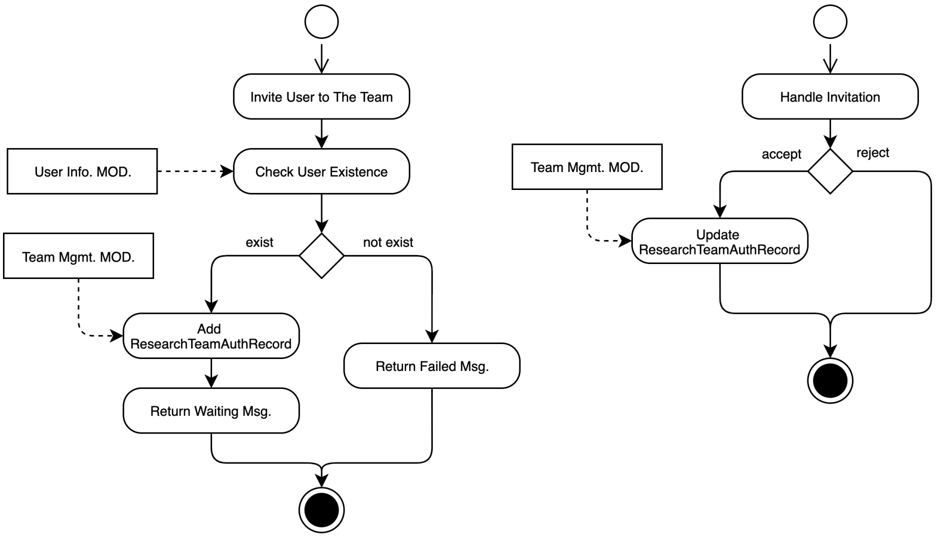


Figure 2.17: Activity Diagram for SRS 6.1.2

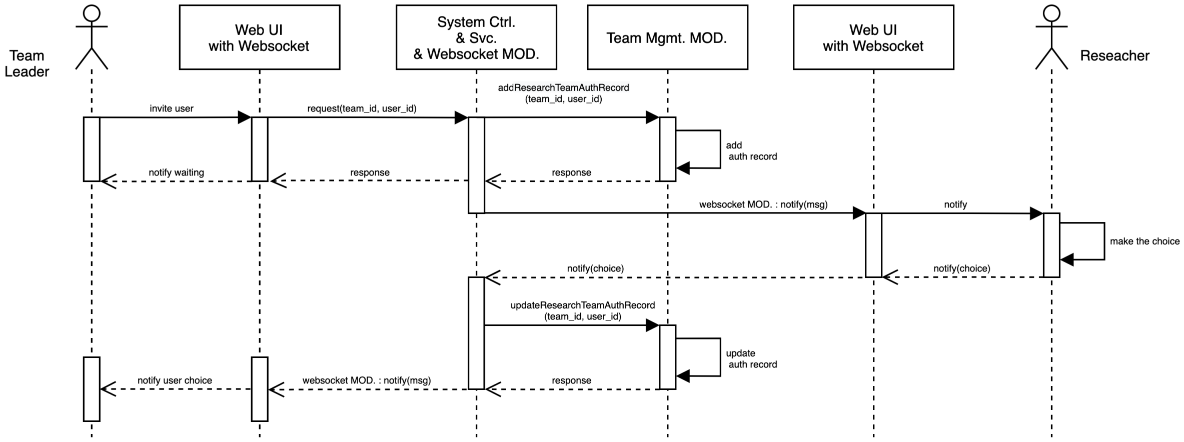


Figure 2.18: Sequence Diagram for SRS 6.1.2

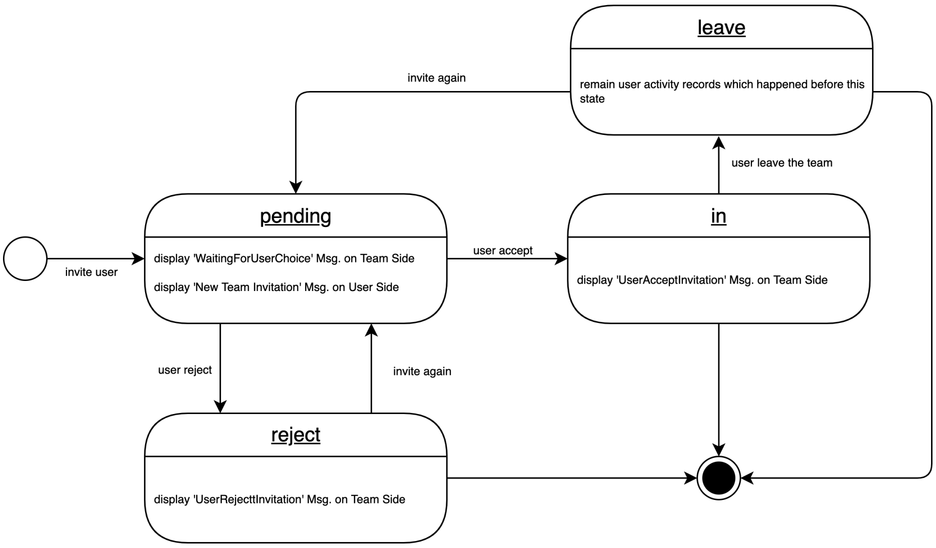

leave
remain user activity records which happened before this state
invite again
user leave the team
pending
display 'WaitingForUserChoice' Msg. on Team Side
display 'New Team Invitation' Msg. on User Side
invite user
user accept
in
display 'UserAcceptInvitation' Msg. on Team Side
user reject
invite again
reject
display 'UserRejecttInvitation' Msg. on Team Side

Figure 2.19: State Diagram for SRS 6.1.2

3 Functional Requirements Specification

3.1 Conventions for All API

Every HTTP responses from back-end server and the requests from front-end shall follow the conventions presented below.

Conventions for HTTP response format

1 Response shall format with 'application/json' content-type;

2 Response json shall contain a property "code";

3 The value of the "code" shall follow those rules:
- Integer type;
- '0' stands for the request is succeed in both logical and business aspect;
- Other codes stands for the request is failed;
- Specific code shall be explained in every FRS;

Conventions for API Auth.

The system should follow the concept of JWT for HTTP request&response authorization. For Python platform, module PyJWT is recommended.

1 The token shall be generated after the ***login*** request is in code '0';

2 The token shall be placed in the HTTP headers with key 'Authorization';

3 Every auth-reqired request shall always verify the token first;

4 The token shall be expired after the ***logout*** request or the expire time;

Other Conventions

1 API url should follow the fixed context path: "[host]:[port]/**scholar-hub**/[request-url]";

2 Only **GET** and **POST** request are allowed;

3 Every data of the **POST** shall be placed at the "multipart/form-data" field;

3.2 FRS 1

3.2.1 FRS 1.1 - Login Module

FRS 1.1.1 - Login

Appendix A: Glossary

ICDE: stands for Information Capture and Dissemination Environment.

Researcher: academic researcher with needs a paper search engine to help their research work.

Researche Team: a team of researcher.

Paper: academic papers which stored in the system.

SRS: stands for system requirements specification.

FRS: stands for functional requirements specification.

Appendix B: Editing Records

- **v1.0** Jun Huang - 03/11/2021